5-25-2023

Predicting flux data for exoplanet detection.

Ronald Kroening

PREDICTING FLUX DATA FOR EXOPLANET DETECTION

Ronald Kroening, B.S. in Computer Science

Dr. Francis Parisi, Thesis Advisor

Pace University

ABSTRACT

This paper will focus on utilizing five different methods of machine learning models to properly classify celestial bodies orbiting a star as an exoplanet or a false positive. We will be utilizing a recurrent neural network (RNN), a logistical regression model (LR), and a Random Forest Classifier (RF). The focus of the data set was to improve access to balanced data in the form of extracted features and time series graphs, as well as looking into potential solutions for previous shortcomings outlined in prior work, specifically relating to logistical regression models. Training data was assembled from Astronet, a pipeline that included data from the Kepler satellite and the transiting exoplanet survey satellite (TESS). 6,000 different stars were analyzed, with 3,000 being systems without any exoplanets confirmed, and 3,000 being confirmed instances of an exoplanet. The random forest model was optimized with GridSearch for performance, as well as one of the logistical regression models. Two logistic regression models were also used, with logistical regression as the base and a C4.5 Decision Tree and Support Vector Machine used as different models stacked on top of the predictions, respectfully. In addition, a recurrent neural network was also trained on a smaller subset of the data to account for the necessary computational power, with the class split remaining the same. The metrics used for analysis were accuracy, precision, recall, F1 score, and area under the receiver operator characteristic curve (AUCROC). The results showed all machine learning models trained having at least 90% accuracy, with improvements shown in logistical regression from the stacking methods used. The results also showed the benefits of using LSTM to optimize recurrent neural networks to solve the disappearing gradient problem.

Table of Contents

Introduction

Being able to detect an exoplanet is a crucial part of humanity's longstanding journey to understand space. Analyzing other planets like our own can help us look at our planet in new ways, as well as learn more about the universe. It is also one of the few examples in which the answers are literally written in the stars, as flux analysis is one of the most time tested methods of detecting an exoplanet (*What's a Transit? – Exoplanet Exploration: Planets Beyond Our Solar System*, 2020.). In this context, flux is defined as how bright a star appears at one point in time as it is perceived by another object. In this case, flux would mean how bright a star appears from the perspective of a satellite analyzing it. When the flux decreases dramatically, this is a strong suggestion that the star is harboring an exoplanet, as the drop and apparent brightness can be explained by an exoplanet crossing in front of it (*Down in Front!: The Transit Photometry Method*, 2020). Repeated instances of this occurring are a strong signal of an orbit occurring. In order to draw conclusions about this data, graphs of the flux over a set period of time are used from the databases mentioned before. The given models are then run on the graphs in order to look for patterns common with the presence of an exoplanet. (Thomson, 2015).

Literature Review

Background

The history of exoplanet discovery is actually very short. It was only in 1983 when the first planet outside our solar system was discovered in the form of a planetary disc. Done with a telescope measuring 2.5 meters long. This development did further investigations from space related agencies around the world, and it was in 1992 that the first ever exoplanets were discovered over 1000 light years away from earth (NASA, 2022.). It existed in a star system akin to the ones used in this research project, and they were observed orbiting a star in the Virgo constellation. Eventually, the first ever planet was discovered using the transit method (the one covered in this paper) in the Pegasus constellation.

The Kepler satellites that were used for this data set were launched in 2009, finding its first exoplanet just under two years later. Due to a malfunction it had to and its mission in 2013, but started again in 2014 and continued until October 30th, 2018. Its data was processed by NASA and stored in numerous databases, such as MAST- the Milkulski Archive for Space Travel, and TESS- the Transiting Exoplanet Survey Satellite (MAST, 2020). The data contains information about different celestial bodies such as their size, coordinates, and relative age, as well as information in the form of images. In addition to this, certain celestial bodies have a type of data known as a light curve. A light curve is a measurement over an extended period of time focused on analyzing how bright an object appears to be relative to the measuring body, in this case the satellites. (NASA, 2013.) This is a value known as the flux, which is what these light curves measure. A common object in space that has this type of data are stars, where satellites

can measure how bright they are over an extended period of time. In the universe, there are certain types of stars that are known to be planetary candidates (Batalha, 2014). These are stars that are confirmed to have at least one exoplanet orbiting them, similar to our Sun having planets orbiting it. The purpose of determining whether a star has an exoplanet is that it allows us to not only expand our knowledge about the universe, but also use the information about the exoplanets to determine important facts about our Earth. One of the difficulties with this task is that the distance of these stars makes it difficult to send aircraft to them in order to get physical confirmation of the presence of exoplanets. However, thanks to important methods in data processing, we do not have to actually see the planet in order to know if the star contains an exoplanet. One of the most popular and effective methods is by utilizing the light curve mentioned earlier, that being the apparent brightness of a star over an extended period of time, combined with a method known as the transit method. The transit method is based on the idea that any exoplanet, like any celestial body orbiting a star in a system, would orbit around the star. If we had a satellite facing the star and analyzing at, this means that there would be periods in the light curve during which the exoplanet would be passing in front of the star (Down in Front!: The Transit Photometry Method, 2020). This illustrates something called a transit signal, which is a dramatic drop in the star's apparent brightness which indicates the moment at which an exoplanet is passing in front of the star. This method allows for us to train machine learning algorithms on light curve data. By training a model to identify those transit signals, we can train a model that can make mostly accurate predictions of whether a light curve represents an exoplanet. It is important to note, and this is something that is noted in the original data sets, that the star systems that contain exoplanets are noted as planetary candidates, meaning they are candidates for potentially containing an exoplanet-we would need further data to confirm as

there could be other reasons for the drops in the light curve that would usually result from an exoplanet, a common one being binary stars, which are stars that orbit a star in a system that can mimic the presence of an exoplanet. (Institute of Astrophysics of the Canary Islands, 2002)

Analysis of Methods

Prior to the transit method, the common method for detecting exoplanets was through their radio frequencies (Wenz, 2019). Each star that would host an exoplanet would give off radio frequencies in the form of pulses (which was why neutron stars can be known as pulsars), With the exoplanet signal coming from differences in the frequency- if a star's frequency suddenly changed, it would imply the presence of an exoplanet. In addition to its existence, the observation also allowed it to determine that the planet may contain important elements to suit life, such as water, oxygen, and nitrogen (it's distance two it's host star would make this impossible as its atmosphere was deteriorating due to the star). The first ever confirmed exoplanet was discovered using a technique called radial velocity. This was based on the idea that as a star in a solar system orbits its center of mass it wobbles, in a way that is affected by the amount of planets in the star system. This is viewable through a spectral graph, which was able to be analyzed in order to attract a star's movement, leading to the discovery of the exoplanet (NASA & Brown, 2022).

While radio frequencies are still a viable method of detecting exoplanets, the abundance of light curve data available as well as the format of common machine learning algorithms makes the light curve approach more convenient, while still maintaining reliability of detection. The core focus of the research being conducted this training machine learning algorithms to

apply the transit method to light curve data. As mentioned before, the transit method involves focusing on specific dips in light curves, with these dips coinciding with the presence of an exoplanet passing in front of a star and satellite. These light curves appear in time series format, a time series being a graph in which the x axis are moments in time and the y axis is the apparent brightness at that moment in time. For example, given the following graph:

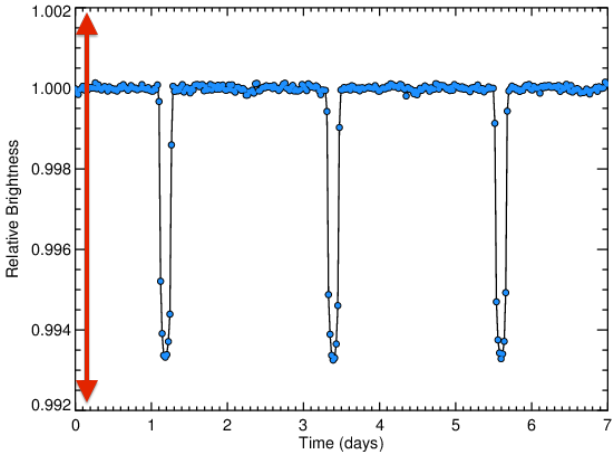

Image Citation: (Vanderburg, 2021.)

This graph is provided from a transit light curve tutorial from MIT, and it illustrates how the transit signals are shown in a given light curve. While value wise they may not seem that different, in proportion to the rest of the graph, it represents a significant deviation. It is also important to note that these transit signals can also be impacted by the size of the exoplanet orbiting the star and the size of the star itself, as larger planets can lead to more significant drops in the light curve. Now that we have the method, the next question to answer is how exactly we will train the machine learning algorithms to understand the curve, and the answer relies on the specific machine learning area that you want to focus on. It is possible to use similar principles from image recognition, as the light curve could be stored as an image or even represented as one and processed through models that have been proven trustworthy in classifying images (Yi,

2019). Certain models, such as decision trees and statistical based models, perform better when analyzing features rather than the actual light curve. For this, you would have a set of features that you would want to extract from the given model, such as average brightness and standard deviation of brightness. These features would then be analyzed by a model that would trained itself to understand which features are the most significant in analyzing if a light curve contained an exoplanet or not-this could prove to be more efficient rather than handling the actual light curve which could be significantly large and take up a large amount of computing power to process (Malik, 2021).

Research Framework

For this project, the goal is to focus on logistic regression models, as the references utilized outlined how the logistic regression models performed worse than other models used for exoplanet detection. In addition, it also intends on providing a more in-depth analysis through the use of different model metrics as well as an improved access to exoplanet data with the resources derived from the prior research. These are the models being used as well as a description and history of their research.

Logistic Regression: This type of model uses statistical inference based on a data set of features to create a logistical, or sigmoid, function that could effectively divide the data entries into their respective classes. It takes the extracted features of the light curve into account when it is making its predictions by modeling the relationship between how likely a feature is able to predict the class of a specific entry. It has been proven to be effective with these types of problems but struggles with detailing more complex relationships that leads to a loss in performance.

Random Forest: In most of the papers studied, this model was consistently the highest performing model tested. Also, a feature-based data set like logistic regression, it instead consists of a number of decision trees, each focused on a different sample of the data set. Each decision tree takes the features of each sample into account, and the sides if a feature is significantly different enough to make predictions on the class of given elements in a dataset. The results of these decision trees are then combined through majority voting, or the average prediction from each tree decides the classification. Given the amount of data it can process, it is able to make very accurate predictions and do so in relatively short time.

Recurrent Neural Network: this is one of the most common models for processing any type of time series data, whether it be light curves or audio files. This is because of the architecture of the recurrent neural network, which is designed so every output in the network is also a weight in the next node in the network (Amidi & Amidi, 2018). With this design, the recurrent neural network allows each individual time stamp to be processed and the information gained from one influence the information gained from the next. This is something commonly seen in problems such as weather forecasting and stock price predictions, where it is important to get prior information to predict what happens next. One of the largest weaknesses of recurrent neural networks derives from this, and it is because this process takes a lot of computing power, augmented by the fact that the nature of it processing each individual time series limits the optimization that a computer can do to train and test on the data set.

The random forest model and recurrent neural network were not modified, as well as one of the logistical regression models, to serve as a control variable. In addition to these three, I implemented two variants of logistic regression that focused on model stacking. This is the

process of using the predictions from one model to inform the decisions of another model. When picking out models to use for each of the variants, I kept in mind that I wanted models that were different from a logistical regression model, as I wanted to ensure that they would be able to catch certain trends and abnormalities that a logistical regression model could not cover. One variant of the model had a decision tree, known as a C4.5 decision tree. This decision tree is different from others because it utilizes something called a gain ratio to split a data set by features. The gain ratio is a focus on the information gained by splitting out a feature divided by the number of categories available for the given feature. What this means is that in addition to considering how much information will be gained from focusing on a specific feature, it will also take into account a feature with more categories, which prevents a model from overfitting, or reacting too quickly to implement new information (Mathew, 2022). The other variant used focuses on what is known as a support vector machine (SVM). A support vector machine focuses on analyzing a line or hyperplane that could separate data entries into classes based on a set number of features. One of the weaknesses with this type of model is that it cannot consider as many features (Bennet, 2001). This is the reason why this variant had the support vector machine stacked on top of the logistical regression model, which means that the logistic regression model sent its outputs to the support vector machine. This would allow for the logistical regression model to make its general predictions as to the classes, and allow for these support vector machine to make more specific relations between the entries and fine tune the classifications. The first-ever application of machine learning to detect and confirm exoplanets was made in 2020 by researchers at the University of Warwick. The model was made in collaboration with the Turing institute and utilized the transit method as well as planetary images to classify if a star had an exoplanet (University of Warwick, 2020). What follows is a review of further

applications, utilizing a variety of machine learning algorithms and metrics in order to classify planetary candidates.

Reference Review

A. N. Herur, R. Tajmohamed and J. G. Ponsam's work outlined in *Exploring Exoplanets using kNN, Logistic Regression and Decision Trees* focused on utilizing Kepler data in order to explore exoplanet data for binary classification. The goal was to be able to analyze a given star and attacked whether it had an exoplanet. It did so utilizing decision trees, logistical regression, and a k-nearest neighbor model. The purpose of using this paper was to understand the methods that had been traditionally used to solve this problem. It showed the effectiveness of decision trees as well as the shortcomings of logistical regression. While it also showed the promise of K nearest neighbors in order to classify exoplanets. The use of a decision tree with a logistical regression model was something noteworthy as the common applications were to use a random forest, which is a large amount of decision trees. This was taken into account when consider the architecture logistical regression model and ended up using this as part of the project.

Malik et. al's work outlined in *Exoplanet Detection Using Machine Learning* used a gradient boosted tree to classify stars as having an exoplanet or not. It also detailed important documentation to a data set that holds light curves from the Kepler satellite. The paper not only showed the effectiveness of different models to classify exoplanet data, it also outlined the process of feature engineering, as well as certain packages that are used to operate on the exoplanet data. One interesting feature of their data set was their reliance on simulated data rather than the exoplanet data set. In order to account for gaps in the time series data, They had transit signals manually injected at certain points, due to the fact that there was a lack of light curve data for stars with exoplanets. This paper also showed the benefits of automatic feature

extraction for exoplanet detection, as these features are able to be trained on a model without regard to the time.

Margarita Bugueno et. al's work outlined in *Refining Exoplanet Detection Using Supervised Learning and Feature Engineering* showed a new method of feature engineering in which the dataset is optimized in order to improve model performance, and used k-nearest neighbors, Logistic Regression, and Random Forest in its classification. The use of Principal Component Analysis for reducing the dimensionality of the dataset for feature engineering was crucial to the experiment shown as it proved to be an effective measure in the experiment It also made known the variety of time series specific feature engineering techniques, as well as a further dive into the metrics used to evaluate the models.

Dr. C.J Shallue & Dr. A. Vanderburg's work outlined in *Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90* focused on utilizing machine learning techniques for exoplanet detect but focused specifically on two different systems. It showed more specific algorithms for analyzing star systems, as well as the debut of Astronet, an innovative pipeline for kepler data. The use of Astronet aside from the design of machine learning models served as a way to improve the amount of data known about exoplanets as it provided a direct connection to the kepler dataset.

Miguel Jara-Maldonado et. a's work outlined in *Transiting Exoplanet Discovery Using Machine Learning Techniques: A Survey* served as a metaanalysis on machine learning algorithms used to detect exoplanets, including the data and the models themselves. It showed the fundamental areas of focus for those looking to research this problem, including the underlying methods used by machine learning algorithms for exoplanet detection. The model diversity that were experimented with, from Least Square Models to Random Forests and

MultiLayer Perceptrons, proved to be useful in not only building successful models but also in outlining a format for exoplanet research. It was an introduction to the problem at hand- it introduced me to the concept of the transit method, which was the most effective method for machine learning algorithms to process for exoplanet detection as well as important benchmarks for the models used on the dataset.

Conclusion

The goals of the research are to streamline the process of doing future research by organizing a balanced data set of light curve data as well as providing pre extracted features that can be trained on future machine learning models. It also builds upon prior research by showing the strength of the tools commonly used for time series analysis, such as TSfresh, and the reliability of the newer system of astronet as a pipeline for organizing a data set. In addition, this research will also build on prior weaknesses in machine learning models that have been specifically used to solve this problem and comparing it against models that have done predominantly well.

Dataset

Looking for common data sets on space exploration is difficult, one reason being that the archives where the data is eventually stored are large and difficult to navigate. Publicly available datasets on common websites do not prove to be as insightful and contain as much information as the project calls for. I assembled this dataset from kepler.io, a Python pipeline that connects to Kepler-II data including light curves for different stars and whether or not they are planetary candidates, from the Google Research project exoplanet-ml (Shallue, 2022). The current data set contains a column for the following:

- a binary classification of each star system, a zero for no exoplanet and a 1 to denote the presence of at least 1 exoplanet.
- A column for the flux at the xth timestamp, where x is a number from 0 to 4160. 4160 is the largest timestamp for each star in the dataset, chosen because it is the largest size possible for a subset of each dataset, best explained by the following formula:

$$(S),\ S = (s[i])]$$

Where s[i] denotes the ith flux graph in each star s. The stars for the dataset are chosen by iterating through the available stars and choosing the first 6000 that fulfill the classes evenly- the first 3000 not planetary candidates and the first 3000 with exoplanets. For each star chosen, the largest flux graphs available are added to the dataset, which are then cut down so that they are all the same size for feature engineering. The dataset utilized a 50-50 split between each of the binary classes, with a total of 6,000 entries in the dataset. For the different sets, 80% of the data was used for training, with 10% being set aside each for testing and validation sets. After the data was processed, any remaining values were interpolated on, an algorithm used to film empty spots and data utilizing these surrounding data for that entry.

Specifically for these feature-based models in logistical repression as well as the random forest model, we will be using feature engineering in order to extract relevant features from the data set. This was done utilizing tsfresh, a package designed for extracting a large amount of features from a time series. In contrast due to its architecture, the recurrent neural network did not utilize the features and strictly used the time series as input. At the time of model testing, tsfresh could not be used on the newest version of Python- one of the packages it relies on was not supported, meaning that tsfresh was run on a lower version of Python. It was able to extract 738 different features from all stars in the dataset, which were stored in a CSV file. As for the RNN data, two CSV files were used: one stores the flux data for each star, and the other stores the timestamps for each star. This was done for convenience, as each star had different timestamps that needed to be accounted for. The types of features taken were various, ranging from normal data features such as mean and standard deviation, to more graph specific features such as longest strike below mean, referring to the longest period in the graph where the data points were significantly below the mean. This would be important in identifying transit signals, making it one of the more important features to be analyzed.

Experiment Design

Recurrent Neural Network

A Recurrent Neural Network is a machine learning model designed for data where order is key. It takes in an input, applies weight, and then loops on itself for a set amount of time, where the weighted input of one neuron in the network becomes the weight of the next neuron. The fundamental aspect of this network is that it is able to use the previous predictions in its neurons to inform the output of the next neuron, with what is known as a recurrent unit. In classification methods, an activation function is applied at the end to deliver a value. The emphasis on order makes it optimal for time series data, where each input is a flux reading at a set timestamp. It is commonly found in problems such as sound recognition and natural language processing, where the elements that precede and succeed others in a graph combined could provide useful information about the potential class.

Logistic Regression

Logistic regression is a method of classification that utilizes the sigmoid function to provide and appropriate distinction between different classes. Given variables, it predicts the probability of the input data being of one class or another based off the probability of the variable given. It has shown to be successful in classification but is still hindered compared to others. This is because it relies on the input data being compressed to a small amount of variables that could end up missing key elements in the data.

Random Forest

Random forest algorithm is common in classification problems due to its efficiency and proven track record in detecting true positives and true negatives (Malik, 2021). Its efforts focus on two numbers: k and n, where k represents the number of decision trees used and n is the number of items from the data set taken to fuel each tree. At the start of the algorithm, a total of k decision trees are generated, and each decision tree takes n samples from the data set as part of its training data. Each decision tree is then trained on the data and gives an output. The results from each the decision tree are then combined to deliver an end result. It is run only once but for these types of problems has a high accuracy rate due to its massive amount of data processing. (Scikit-Learn, 2023).

Methodology

The first focus for this experiment is to test a dataset on models common for timeseries analysis and make it publicly available, the reason being that it is very difficult to find and process data for this specific problem. The second focus is to test different methods to optimize each model with focuses on different weaknesses. The following describes the model weakness that will be explored and the hyperparameters for each model.

1. Random Forest- Grid Search optimization, with C4.5 Decision trees used.
2. Logistical Regression- Testing stacking model with C4.5 Decision tree and SVM models to improve accuracy.
3. RNN- Using LSTM to counter overfitting and dynamic learning rate.

For the logistic regression model, the decision on what models to stack on top of the logistical regression model was made based on model architecture. Typically, it is optimal to use models that are different from the original, and given the weakness of logistic regression being a decreased ability to find relationships on a larger data set, three models were chosen based on their ability to work with large data sets and cover these potential weaknesses. The first is a VLR, or vanilla logistic regression model, meaning a standard logistic regression model with grid search provided for hyperparameters. For the other two models, one will be a logistical regression model stacked with a C4.5 decision tree (chosen for its success with large data sets) and a support vector machine (chosen for its ability to draw integrated relationships between features).

For the recurrent neural network, one of the difficulties with this type of model is that it takes a long time to run. For this, I used a subset of the original data set, focusing on 1000 different

stars. The class split still remained the same at 50% in each. A common problem with recurrent neural networks is what is known as the disappearing gradient problem, which is when during training, the gradient reduces to such a low value that it is no longer able to update the model weights, hindering the learning process (Noh, 2021). For this, the layers in the recurrent neural network utilize long short term memory units (LSTM), would allow for the models to store gradients for future use in order to control its rate of change.

Execution and Results

The metrics used to measure the models are the following (Erickson, 2021):

Accuracy: a measurement of the percent of inputs predicted correctly. It is the sum of all corrected inputs divided by the total number of inputs.

Precision: the measurements of inputs correctly predicted as positive, in this case the stars predicted as correctly having exoplanets. This is denoted as the sum of all the corrected inputs that have exoplanets divided by the sum of every input predicted as having an exoplanet whether correct or not.

Recall: similar to precision, except it is the fraction of how many inputs were predicted as exoplanets of how many of the inputs were actually exoplanets. This is shown as the sum of all inputs correctly predicted to have exoplanets divided by the sum of both the inputs correctly predicted as having exoplanets and those who should have been predicted as having exoplanets.

F-1 Score: this metric is meant to provide a balanced measure of a model's performance and is the mean of a model’s precision and recall

AUC-ROC: an acronym for area under the receiver operating characteristic curve, it measures how a model performs across all potential types of classifications by plotting the rate of true positives against the rate of false positives, and with the resulting metric being the area under that curve.

Loss: also known as a loss function or cost function, loss is a measure of how well the model's predictions align with the actual or expected values. It shows the difference between predicted and target values, with the goal of a machine learning model is to minimize the loss function, as a lower value indicates better alignment between predictions and targets.

Metric	Model	Accuracy	Precision	Recall	F1-score	AUC-ROC	Loss
F-1 Score	VLR	90.68%	98.74%	81.94%	89.56%	90.48%	9.24%
	LR-SVM	90.85%	**99.58%**	81.60%	89.69%	90.63%	9.15%
	LR-C4.5	90.34%	98.13%	81.77%	89.20%	90.14%	9.58%
	RandomForest	91.61%	**99.58%**	83.16%	90.63%	91.41%	8.39%
	RNN_AM_BCE	**100.00%**	85.09%	83.50%	91.01%	91.75%	27.22%
	RNN_AM_MSE	**100.00%**	84.55%	**84.11%**	**91.37%**	**92.06%**	**7.67%**
	RNN_A_BCE	**100.00%**	87.10%	82.61%	90.48%	91.30%	27.05%
	RNN_A_MSE	**100.00%**	84.48%	82.35%	90.32%	91.18%	7.79%
	VLR	90.68%	98.74%	81.94%	89.56%	90.48%	9.32%

In this graph, the first two columns represent the metric used to train the model and the name of the model. The other categories represent the metrics gathered on the test dataset. From this data, we can see that the recurrent neural networks proved to be performing higher than other models, and the random forest as well. In addition, it shows that the unchanged Logistic Regression model (denoted as VLR or Vanilla Logistic Regression) was outperformed by one of its variants, specifically the variant stacked with SVM, which also outperformed every other model on precision when trained with the F1-Scores and AUCROC metrics. The variant stacked with the C4.5 Decision Tree, conversely, was either tied with or outperformed by the VLR at every metric. Out of the recurrent neural networks, the models trained with an MSE loss function tended to perform worse across all metrics than those trained with a BCE loss function, though recorded lower losses.

Conclusion

The concept of being able to train a machine learning model utilizing machine learning methods was achieved thoroughly in this experiment. In addition, the findings of this paper not only showed that logistical regression can be even more useful for time series analysis when it is stacked, but also showed the strength of a larger dataset. This paper will make the data set used publicly available so that more people can test and learn from this experiment, as the process of finding and generating the information needed to run the models was the foundations of conducting this experiment. There were certain limitations on this project that could be overcome, such as the computational power needed to run the recurrent neural network, which resulted in a smaller data set to be used.

Future Work

There are still certain problems involving the use of the transit system that need to be further examined. One of which can be done by adding an additional category to the stars mentioned for binary star systems. Binary star systems are stars in which they are orbited by another star known as a binary star instead of a planet. This system appears like a planetary candidate, especially when the transit method is used, but there are still significant differences. In addition, being able to have more information about the stars, such as their coordinate based location and radius would allow for different features to be generated (Kopparapu, 2018). In addition, this also shows the need for more research in recurrent neural network optimization. The Recurrent Neural Network performed the best out of all models tested, and yet it was limited by the computational resources needed. This is an issue that goes beyond time series analysis, as recurrent neural networks are useful in multiple fields that require data to be processed in time stamps such as video analysis and text generation.

Works Cited

A. N. Herur, R. Tajmohamed and J. G. Ponsam, "Exploring Exoplanets using kNN, Logistic Regression and Decision Trees," 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), Chennai, India, 2022, pp. 1-7, doi: 10.1109/ICSES55317.2022.9914278.

"Exploring Exoplanets Using KNN, Logistic Regression and Decision Trees." IEEE Xplore, https://xplorestaging.ieee.org/document/9914278.

Malik, Abhishek, et al. "Exoplanet Detection Using Machine Learning." ArXiv.org, 5 Mar. 2021, https://arxiv.org/abs/2011.14135v2.

Margarita Bugueno, et. al. Refining Exoplanet Detection Using Supervised Learning and Feature Engineering, https://www.researchgate.net/profile/Francisco-Mena-3/publication/334992434_Refining_Exoplanet_Detection_Using_Supervised_Learning_and_Feature_Engineering/links/5fa17338458515b7cfb5eca0/Refining-Exoplanet-Detection-Using-Supervised-Learning-and-Feature-Engineering.pdf.

Shallue, C. J., & Vanderburg, A. (2018). Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90. The Astronomical Journal, 155(2), 94.

References

Amidi, A., & Amidi, S. (2018). CS 230 - Recurrent Neural Networks Cheatsheet. From https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

Baidoo, E. (2020). An Analysis of Accuracy using Logistic Regression and Time Series. *Grey Literature from PhD Candidates.*, *2*. https://digitalcommons.kennesaw.edu/cgi/viewcontent.cgi?article=1001&context=dataphdgreylit

Batalha, N. M. (2014). Exploring exoplanet populations with NASA's Kepler Mission. *PNAS*, *111*(35), 12647-12654.

Bennet, K. P. (2001). Support vector machines: hype or hallelujah? *ACM SIGKDD*, *10*(1145). https://doi.org/10.1145/380995.380999

Down in Front!: The Transit Photometry Method. (2020). The Planetary Society. From https://www.planetary.org/articles/down-in-front-the-transit-photometry-method

Erickson, B. J. (2021). Magician's Corner: 9. Performance Metrics for Machine Learning Models. *RSNA*, *3*(3). https://pubs.rsna.org/doi/full/10.1148/ryai.2021200126

Institute of Astrophysics of the Canary Islands. (2002, March 25). The Transit Method. From http://research.iac.es/proyecto/tep//transitmet.html

Kopparapu, R. K. (2018). The American Astronomical Society, find out more The Institute of Physics, find out more Exoplanet Classification and Yield Estimates for Direct Imaging Missions. *The American Astronomical Society, find out more The Institute of Physics, find out more Exoplanet Classification and Yield Estimates for Direct Imaging Missions*, *856*(2). https://iopscience.iop.org/article/10.3847/1538-4357/aab205

Malik, A. (2021). Exoplanet Detection with Machine Learning. *2011*(14315). https://arxiv.org/pdf/2011.14135.pdf

MAST. (2023.). MAST Home. From https://archive.stsci.edu

Mathew, P. A. (2022, January 7). *C4.5 Decision Tree. Explained from bottom up | by Praveen Alex Mathew*. Level Up Coding. From https://levelup.gitconnected.com/c4-5-decision-tree-explained-from-bottom-up-67468c1619a7

NASA. (2023.). *Light Curves - Introduction*. Imagine the Universe! From https://imagine.gsfc.nasa.gov/science/toolbox/timing1.html

NASA. (2018, October 30). *Mission overview*. NASA. From https://www.nasa.gov/mission_pages/kepler/overview/index.html

NASA & Brown, T. (2023.). *Historic Timeline | Explore – Exoplanet Exploration: Planets Beyond our Solar System*. Exoplanet Exploration. From https://exoplanets.nasa.gov/alien-worlds/historic-timeline/#age-of-discovery-5-000-exoplanets

Noh, S.-H. (2021). Analysis of Gradient Vanishing of RNNs and Performance Comparison. *MDPI*, *12*(11). https://www.mdpi.com/2078-2489/12/11/442

Radial Velocity. (2022.). National Schools' Observatory. From https://www.schoolsobservatory.org/learn/astro/exoplanets/detection_methods/rv

Scikit-Learn. (2023.). *sklearn.ensemble.RandomForestClassifier — scikit-learn 1.2.2 documentation*. Scikit-learn. From https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.predict

Shallue, C. (2022, February 23). *AstroNet: A Neural Network for Identifying Exoplanets in Light Curves*. YouTube. From https://github.com/google-research/exoplanet-ml/tree/master/exoplanet-ml/astronet

Thomson, S. E. (2015). A MACHINE LEARNING TECHNIQUE TO IDENTIFY TRANSIT SHAPED SIGNALS. *The Astrophysical Journal*, *812*(46). https://iopscience.iop.org/article/10.1088/0004-637X/812/1/46/pdf

University of Warwick. (2020, November 28). *First ever exoplanet discovery made by artificial intelligence*. Warp News. From https://www.warpnews.org/space/first-ever-exoplanet-discovery-made-by-artificial-intelligence/

Vanderburg, A. (2018.). *Transit Light Curve Tutorial*. Andrew Vanderburg. From https://avanderburg.github.io/tutorial/tutorial.html

Wenz, J. (2019, October 8). *How the first exoplanets were discovered | Astronomy.com*. Astronomy Magazine. From https://astronomy.com/news/2019/10/how-the-first-exoplanets-were-discovered

What's a transit? – Exoplanet Exploration: Planets Beyond our Solar System. (2020.). Exoplanet Exploration. From https://exoplanets.nasa.gov/faq/31/whats-a-transit/

Yi, M. (2019). Robust Deep Graph Based Learning for Binary Classification. *IEEE*, *1912*(332). https://arxiv.org/pdf/1912.03321.pdf